Introduction to the Streaming Data Analytics and Applications Mini-track

Mehmed Kantardzic, Ph.D.
Computer Engineering and Computer Science Department
J.B. Speed School of Engineering
University of Louisville
Louisville, KY 40292, USA
mmkant01@exchange.louisville.edu

Jozef Zurada, Ph.D., D.Sc.
Computer Information Systems Department
College of Business
University of Louisville
Louisville, KY 40292, USA
jmzura01@louisville.edu

Streaming data can be considered as one of the main sources of what is recently called big data. Large volumes of data streams are generated by numerous real-world applications, from real-time surveillance systems and on-line transactions in financial markets, to electric power grids with remote sensors and industry production processes. Data streams are time dependent, fast changing, massive, and potentially infinite. Many challenges in streaming data analytics still exist, and researchers are highlighting problems such as: protecting data privacy, dealing with legacy systems, handling incomplete and delayed information, analyses of complex data, or evaluation of stream mining algorithms in time. It is recognized that for effective processing and analyses of streaming data, new data structures, techniques, and algorithms are needed. This mini-track contains two papers relevant to streaming data analytics and applications.

Most algorithms developed for evolving data streams make simplifying assumptions on the timing and availability of information. In particular, they assume that information is complete, immediately available, and received passively and for free. These assumptions often do not hold in real-world applications. The problem is especially recognized in the streaming data classification frameworks, where changes in data require immediate adjustments in a predictive model. A new classifier is based on the assumption that labeled samples are available when necessary, and any delay in labels may delay model adjustment, and therefore reduce performances of a framework. In the first paper titled "Sliding Reservoir Approach for Delayed Labeling in Streaming Data Classification" the authors present a new framework, which gives a robust solution for delayed labeling problem. The proposed Sliding Reservoir Approach for Delayed Labeling (SRADL) framework contains three main components: the label reservoir that continuously keeps track of the arrival of delayed labels, the change detection module that monitors a concept drift, and the semi-supervised learning module that updates the framework's predictive models. Models are built when necessary based on available labels in the reservoir, without waiting for all required labels for changing stream. Experimental results with spam streaming data show that the proposed SRADL approach significantly improves classification results (by 7.5%) compared with traditional approaches where the frameworks are waiting for all required labels and then building new models.

In the second paper titled "The RADStack: Open Source Lambda Architecture for Interactive Analytics", the authors present the design principles and the architecture of an open-source data analytics stack for fast, flexible, and low-latency analytic queries on near real-time data. The authors also discuss the methods of providing interactive analytics and a flexible data processing environment to handle a variety of real-world workloads. The RADStack is a collection of complementary technologies used together to power interactive analytic applications designed to overcome the limitations of pure batch processors and pure stream processors. The key pieces of the RADStack are: Apache Kafka, Apache Samza, Apache Hadoop, and Druid. Druid, a serving layer, is designed for interactive exploratory analytics and is optimized for low latency data exploration, aggregation, and ingestion, and is well suited for OLAP workflows. Samza and Hadoop complement Druid and add data processing functionality, and Kafka enables high throughput event delivery problem. This combination of technologies is flexible enough to handle a wide variety of processing requirements and query loads. The presented system seamlessly returns best-effort results on very recent data combined with guaranteed-correct results on older data. The key contributions of the paper include the architecture of the stack itself, the introduction of Druid as a serving layer, and the model for unifying real-time and historical workflow.

URI: http://hdl.handle.net/10125/41357
ISBN: 978-0-9981331-0-2